Traversing algorithms

- sequential polling algorithm
- traversing connected networks (tree construction)
  - complexity measures
  - tree terminology
  - tarry's algorithm
  - awerbuch's algorithm

What are traversal algorithm

- Traversal algorithm is a wave algorithm with the following properties:
  - each computation contains one initiator which starts computation by sending one message
  - when a process receives a message it either sends out one message or decides
  - traversal algorithm can be viewed as follows: there is a single token that "visits" all processes
  - we've already looked at one traversal algorithm what is it?

Sequential polling algorithm

```
var req, reply, task 0; (* for initiator only *)

For the initiator:
  (* Write SeqPh = {q0, q1, …, qn-1} *)
  begin while req < #SeqPh do
    begin send (tok) to q0, …, qn-1; (* message (tok); reply = reply + 1 *)
      while end ;
      decide
    end ;
  end;

For non-initiators:
  begin move (tok) from q; send (tok) to q end
```

- Same as polling only visiting of nodes is done in sequence rather than in parallel
- is it a traversal algorithm?
- how many messages are sent in the algorithm?

Measuring efficiency of algorithms

- Time complexity. Time is idealized
  - a process executes internal events in zero time
  - one time unit passes from the moment the message is sent until it is available for receipt
- Message complexity - number of messages it takes the algorithm to carry out specified task
- for traversal algorithms time complexity is equal to message complexity. Why?
- Is it true for all wave algorithms?

Tree terminology

- Spanning tree - a tree that contains all nodes of the network
- leaf - a node that has just one incident edge in the spanning tree
- rooted tree - a tree that has one distinguished node called root
- node’s ancestor - a node that lies on the path from this node to the root
- father - closest ancestor
- node’s descendant - a node that lies on the path from the node to a leaf
- child - closest descendant
- frond edge - edge that is not in spanning tree
- depth-first spanning tree - a tree where a frond edge connects only an ancestor and a descendant

Tarry's algorithm

- Works for arbitrary networks
- initiator forwards token to one of neighbors, each neighbor forwards token to all other nodes and when done returns the token
- a node is free to chose any node to forward token to
- is Tarry's a traversal algorithm?
- does Tarry's algorithm produce a spanning tree?
- is it depth first spanning tree?
- What is the complexity of Tarry's algorithm
  - $2E$
Classical depth-first search

- a restriction of tarry's algorithm
- if a node gets a message over a frond edge it immediately forwards it back
- is the tree constructed a depth-first tree?

Awerbuch's depth-first search

- does spanning tree construction in time proportional to the number of nodes (linear time)
- prevents token forwarding over frond edges - each process knows which neighbors were visited before it forwards the token
- node notifies neighbors that it is visited by sending <vis> messages to them
- time complexity – 4N-2 (token traverses N-1 edges twice and is delayed at every root node for two time units)
- message complexity – 4E (<vis> and <ack> is sent along each frond edge twice, <vis> from father to son, <ack> - back, <tok> - twice along each tree edge)